REMARKS

As an initial matter, it appears that the Office did not consider the Preliminary Amendment that was filed in the subject application on March 14, 2003. In particular, the Office Action Summary indicates that the subject Official Action was responsive to communications filed on 18 October 2001, which is the filing date of the application. The Image File Wrapper for the subject application indicates that the Preliminary Amendment was entered and should have been considered. Having said this, it is anticipated that the Examiner's current rejection would be maintained even if the Preliminary Amendment was considered. Therefore, Applicant addresses the outstanding rejections as if the Preliminary Amendment was considered.

The Examiner objects to the Specification due to an informality. Applicant has amended paragraph [0001] to incorporate the provisional application serial number as suggested by the Examiner. The provisional application serial number was not known at the time of filing the subject application.

The Examiner has also objected to the Declaration for not including the provisional application serial number. As described above, at the time of filing the subject application, the provisional application serial number was not yet known. In order to complete the Declaration, Applicant submits herewith an Application Data Sheet setting forth the provisional application serial number relied upon.

Claims 1-35 remain in this application with claims 1 and 35 in independent form. Claims 1 and 35 have been amended and claims 2-34 remain unchanged. It is noted that the subject application has been granted special status on April 13, 2004 as a result of the Petition to Make Special in accordance with 37 C.F.R. §1.102(d).

Claims 1-35 stand rejected under 35 U.S.C. §102(b) as being anticipated by Hamanaka et al. (United States Patent No. 5,386,566). In particular, the Examiner contends that Hamanaka et al. discloses a method of communicating across an operating system using a plurality of processes (501a, 502a, 503a, 504a) and a plurality of memory sources disposed within one or more processors *(see Figure 24)*. The Office further

contends that the method comprises each of the steps as claimed in the subject application *(see col. 30, lines 41-50 and col. 29 – col. 30).*

Rejection of a claim under 35 U.S.C. §102 requires that each and every limitation be found in the cited reference. If even a single limitation of the rejected claim is not found in the cited reference a rejection under 35 U.S.C. §102 is improper and must be withdrawn.

Applicant respectfully submits that each and every limitation of claims 1 and 35 are not found in Hamanaka et al., either expressly or inherently. Specifically, Hamanaka et al. fails to disclose retrieving an address for a next process to be executed *prior to completing* execution of a prior process. The claim amendments have been made only to further clarify the scope of the invention and thus do not amend substantive limitations of the claims. Accordingly, Applicant traverses the 35 U.S.C. §102 rejection as set forth in greater detail below.

Claim 1

Claim 1 is directed toward a method of communicating across an operating system. The operating system comprises a plurality of nodes with each node having one or more processors capable of executing a plurality of processes and having a plurality of memory sources. It is to be appreciated that the term "operating system", as used in the subject invention, is intended to include the nodes and the associated hardware and software components. Whereas, the term "operating system" as used by some of the prior art (including Hamanaka et al.) generally means software that controls the operation of a computer and directs the processing of the user's programs, such as Microsoft Windows® operating systems. The term "operating system" as used by Hamanaka et al. should not be confused with or analogized to the term "operating system" as used in the subject invention.

The method comprises the steps of detecting an event within the system and ***extracting an initial process address*** from one of the memory sources to determine a location of an initial process in response to detecting the event. Next, ***an initial data address is extracted*** from one of the memory sources to determine a location of initial

data to be used in the initial process in response to detecting the event. Executable code of the initial process located at the initial process address is executed and ***a second process address is extracted*** from one of the memory sources to determine a location of a second process to execute ***prior to the completion of the execution of the executable code*** of the initial process.

Claim 35

Claim 35 is also directed toward a method of communicating across the operating system. As discussed above in relation to claim 1, the operating system comprises the plurality of nodes. The method comprises the steps of detecting an event within the system and ***extracting an initial process address*** for an initial process and ***an initial data address*** for initial data to be used in the initial process in response to detecting the event. The executable code of the initial process is then executed.

The initial data is retrieved from one of the memory sources at the initial data address and the execution of executable code of the initial process with the retrieved initial data is continued to define an initial processed data set. Then, ***an initial processed data address is extracted*** from one of the memory sources and the initial processed data set is written to the initial processed data address.

A second process address is extracted to determine the location of a second process to execute ***prior to the completion of the execution of the executable code of the initial process*** and a second data address is extracted to determine the location of second data to use in the second process. The executable code of the second process is executed and the second data is retrieved from one of the memory sources at the second data address. The execution of executable code of the second process with the retrieved second data is continued to define a second processed data set. ***A second processed data address is extracted*** from one of the memory sources and the second processed data set is written to the second processed data address.

Finally, ***a final process address is extracted*** from one of the memory sources to determine the location of a final process to execute and executable code of the final

process is executed to halt communication of the system until the system detects the event.

In summary, both claims 1 and 35 require the initial process address and the initial data address to be extracted in response to detecting an event within the system. Thereafter, the executable code associated with the initial process is executed. Importantly, ***prior to the completion of the execution of the executable code*** of the initial process, ***a second process address is extracted*** from one of the memory sources to determine a location of a second process to execute. Thus, one objective of the subject invention is achieved, namely, reducing or eliminating delay between communications of different processes on different nodes. The subject invention also optimizes processing time by knowing the next process to be executed prior to completion of the executing process. Said another way, there is almost no delay when executing processes in the various processors since the next process address is known prior to completing the prior process. Therefore, the next process may commence almost immediately upon completing the prior process.

Hamanaka et al.

Hamanaka et al. is directed toward inter-processor communication. In other words, Hamanaka discloses several similar embodiments for facilitating movement of data. Referring to Figure 24, relied upon by the Examiner, multiple processors are illustrated and the disclosure makes clear that the novelty of Hamanaka et al. may be occurring simultaneously in each of the multiple processors. Each of the processors is running an operating system, i.e. software such as Microsoft Windows, that allows data to be organized anywhere in different memory sources. The method disclosed in Hamanaka et al. is achieved by employing protocols that utilize the operating system software to move the data and the process must wait until the data is received to continue executing. In order to send the data, calls, such as software functions, software library calls, and operating system calls, are utilized to send the data between processors of different clusters.

Said another way, Hamanaka et al. organizes the data beforehand, but the movement of the data is accomplished with various degrees of software involvement that must be executed by the processors. Each time the processors execute the software, the transmission of the data is slowed and the overhead of each of the processors is utilized by executing the software. For example, to enable the movement of the data, the processor must respond to instructions. Specifically, the processor requires two sets of instructions, one referred to as "remote store preparation" (RSP) instructions and another referred to as "remote store execution" (RSE) instructions. The processor must communicate with various circuits within the processor to perform the transmission of data.

The processor determines if the instructions are valid and aborts execution of the instructions if the instructions are invalid. In order to determine if the instructions are valid, the processor communicates with the circuits to determine if the instructions have been previously started and that the execution is incomplete. If the instructions have been started, then the processor waits until the execution of the previous instructions are completed. If another instance of the instructions has not been started, only then will the processor execute the instructions. The processor must perform such checks for both the RSP and the RSE instructions to move the data.

The processor of Hamanaka et al. makes extensive of use of protocols, registers, and flags to perform these checks. Protocols are described at col. 12, lines 17-23 and col. 13, lines 34-42 as a communication overhead needed to control the flow of data and continue or stop execution of the process. The protocols rely upon flags to determine if data can be processed *(see col. 14, lines 20-36)*, which means that the processor must wait until the protocols receive the required flags. The delay and waiting of the processor consumes the overhead of the processor and slows the transmission of the data.

In contrast, the subject invention processes the data without waiting or relying upon protocols and flags. The subject invention controls the execution of the processor by extracting the addresses for the next process to be executed prior to the completion of the prior process. Therefore, the subject invention may execute thousand of processes within a short of time *(see paragraph [0031] of the specification as originally field)*.

Hamanaka et al. does not disclose, teach, or suggest such a novel and non-obvious limitation. Unlike Hamanaka et al., the subject invention does not make the processor wait until the flag is received and does not perform multiple checks prior to executing the process. Instead, the subject invention performs the process located at the next process address that was extracted almost immediately upon the completion of the prior process. The subject invention is not concerned with whether the prior process was properly executed or that the data was properly written.

In view of the above, Hamanaka et al. does not disclose, teach, or suggest the step of extracting a second process address for a second process prior to the completion of the execution of the first process. Since Hamanaka et al. does not disclose, teach or suggest each and every limitation as claimed, Applicant believes the 35 U.S.C. §102 rejection should be withdrawn. Thus, claims 1 and 35 are believed to be allowable. Claims 2-34, which depend directly or indirectly from claim 1, are also believed to be allowable.

The remaining references cited but not applied to the claims have been considered. Since the Examiner has apparently considered these references as less pertinent than the above discussed reference, further discussion of the non-applied references, at this time, is considered unnecessary. However, it is respectfully submitted that the claims in the subject application patentably define over all references of record either independently or in combination.

Accordingly, it is respectfully submitted that the Application, as amended, is now presented in condition for allowance, which allowance is respectfully solicited. Other than the extension of time and supplemental information disclosure statement fees, Applicant believes that no fees are due; however, if any become required, the Commissioner is hereby authorized to charge any additional fees or credit any overpayments to Deposit Account 08-2789.

Respectfully submitted

HOWARD & HOWARD ATTORNEYS, P.C.

July 25, 2005
Date

Samuel J. Haidle, Reg. No. 42,619
The Pinehurst Office Center, Suite 101
39400 Woodward Avenue
Bloomfield Hills, Michigan 48304-5151
(248) 645-1483

CERTIFICATE OF EXPRESS MAILING

I hereby certify that the enclosed **Amendment, Three-Month Extension of Time, Application Data Sheet, Supplemental Information Disclosure Statement, PTO/SB/08A, checks for $510.00 and $180.00,** and **return post card** are being deposited with the United States Postal Service as Express Mail, postage prepaid, in an envelope as "Express Mail Post Office to Addressee" Mailing Label No. **EV 612 874 020 US** and addressed to, **Mail Stop Amendment, Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450**, on **July 25, 2005.**

Brenda J. Hughes